Cluster Server Agent for EMC MirrorView Installation and Configuration Guide

AIX, Linux, Solaris

6.2

December 2015

Cluster Server Agent for EMC MirrorView Installation and Configuration Guide

Agent Version: 6.2

Document version: 6.2 Rev 2

Legal Notice

This product may contain third party software for which Veritas is required to provide attribution to the third party (“Third Party Programs”). Some of the Third Party Programs are available under open source or free software licenses. The License Agreement accompanying the Software does not alter any rights or obligations you may have under those open source or free software licenses. Please see the Third Party Legal Notice Appendix to this Documentation or TPIP ReadMe File accompanying this product for more information on the Third Party Programs.

The product described in this document is distributed under licenses restricting its use, copying, distribution, and decompilation/reverse engineering. No part of this document may be reproduced in any form by any means without prior written authorization of Veritas Technologies LLC and its licensors, if any.

THE DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID. VERITAS TECHNOLOGIES LLC SHALL NOT BE LIABLE FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS DOCUMENTATION. THE INFORMATION CONTAINED IN THIS DOCUMENTATION IS SUBJECT TO CHANGE WITHOUT NOTICE.

The Licensed Software and Documentation are deemed to be commercial computer software as defined in FAR 12.212 and subject to restricted rights as defined in FAR Section 52.227-19 "Commercial Computer Software - Restricted Rights" and DFARS 227.7202, et seq. "Commercial Computer Software and Commercial Computer Software Documentation," as applicable, and any successor regulations, whether delivered by Veritas as on premises or hosted services. Any use, modification, reproduction release, performance, display or disclosure of the Licensed Software and Documentation by the U.S. Government shall be solely in accordance with the terms of this Agreement.

Veritas Technologies LLC
500 E Middlefield Road
Mountain View, CA 94043

http://www.veritas.com

Technical Support

Technical Support maintains support centers globally. Technical Support's primary role is to respond to specific queries about product features and functionality. The Technical Support group also creates content for our online Knowledge Base. The Technical Support group works collaboratively with the other functional areas within the company to answer your questions in a timely fashion.

Our support offerings include the following:

- A range of support options that give you the flexibility to select the right amount of service for any size organization
- Telephone and/or Web-based support that provides rapid response and up-to-the-minute information
- Upgrade assurance that delivers software upgrades
- Global support purchased on a regional business hours or 24 hours a day, 7 days a week basis
- Premium service offerings that include Account Management Services

For information about our support offerings, you can visit our website at the following URL:

www.veritas.com/support

All support services will be delivered in accordance with your support agreement and the then-current enterprise technical support policy.

Contacting Technical Support

Customers with a current support agreement may access Technical Support information at the following URL:

www.veritas.com/support

Before contacting Technical Support, make sure you have satisfied the system requirements that are listed in your product documentation. Also, you should be at the computer on which the problem occurred, in case it is necessary to replicate the problem.

When you contact Technical Support, please have the following information available:

- Product release level
- Hardware information
- Available memory, disk space, and NIC information

- Operating system
- Version and patch level
- Network topology
- Router, gateway, and IP address information
- Problem description:
  - Error messages and log files
  - Troubleshooting that was performed before contacting Technical Support
  - Recent software configuration changes and network changes

Licensing and registration

If your product requires registration or a license key, access our technical support Web page at the following URL:

www.veritas.com/support

Customer service

Customer service information is available at the following URL:

www.veritas.com/support

Customer Service is available to assist with non-technical questions, such as the following types of issues:

- Questions regarding product licensing or serialization
- Product registration updates, such as address or name changes
- General product information (features, language availability, local dealers)
- Latest information about product updates and upgrades
- Information about upgrade assurance and support contracts
- Advice about technical support options
- Nontechnical presales questions
- Issues that are related to CD-ROMs, DVDs, or manuals

Support agreement resources

If you want to contact us regarding an existing support agreement, please contact the support agreement administration team for your region as follows:

Worldwide (except Japan)	CustomerCare@veritas.com
Japan	CustomerCare_Japan@veritas.com

Contents

Technical Support 4

Chapter 1 Introducing the agent for EMC MirrorView 9

About the agent for EMC MirrorView 9
Supported software 10
Supported hardware 10
Typical EMC MirrorView setup in a VCS cluster 10
EMC MirrorView agent functions 12
  About the MirrorView agent's online function 13

Chapter 2 Installing and removing the agent for EMC MirrorView 16

Before you install the agent for EMC MirrorView 16
Installing the agent for EMC MirrorView 16
Upgrading the agent for EMC MirrorView 17
Removing the agent for EMC MirrorView 18

Chapter 3 Configuring the agent for EMC MirrorView 19

Configuration concepts for the EMC MirrorView agent 19
  Resource type definition for the MirrorView agent 19
  Attribute definitions for the MirrorView agent 20
  Sample configuration for the MirrorView agent 22
Before you configure the agent for EMC MirrorView 24
  About cluster heartbeats 24
  About configuring system zones in replicated data clusters 24
  About preventing split-brain 25
Configuring the agent for EMC MirrorView 26
  Configuring the agent manually in a global cluster 26
  Configuring the agent manually in a replicated data cluster 27
  Executing the addArrayUser action 27

Chapter 4 Managing and testing clustering support for EMC MirrorView 29

Typical test setup for the EMC MirrorView agent 29
Testing service group migration 30
Testing host failure 31
Performing a disaster test 32
Performing the failback test 32
Failure scenarios for EMC MirrorView 33
    Site disaster 33
    All host or all application failure 34
    Replication link failure 34
    Split-brain in a MirrorView environment 34

Chapter 5 Setting up a fire drill 36

About fire drills 36
Considerations for using MirrorView and SnapView together 37
About the MirrorViewSnap agent 37
    MirrorViewSnap agent functions 38
    Resource type definition for the MirrorViewSnap agent 38
    Attribute definitions for the MirrorViewSnap agent 39
Before you configure the fire drill service group 40
Configuring the fire drill service group 40
    Creating the fire drill service group using Cluster Manager (Java Console) 40
Verifying a successful fire drill 42
Sample configuration for a fire drill service group 42

Index 47

Chapter 1

Introducing the agent for EMC MirrorView

This chapter includes the following topics:

- About the agent for EMC MirrorView
- Supported software
- Supported hardware
- Typical EMC MirrorView setup in a VCS cluster
- EMC MirrorView agent functions

About the agent for EMC MirrorView

The Cluster Server agent for EMC MirrorView provides support for application failover and recovery. The agent provides this support in environments that use MirrorView to replicate data between CLARiiON arrays.

The agent monitors and manages the state of replicated CLARiiON arrays that are attached to VCS nodes. The agent ensures that the system that has the MirrorView resource online also has safe and exclusive access to the configured arrays.

You can use the agent in replicated data clusters and in global clusters that run VCS.

The agent also supports parallel applications, such as Storage Foundation for Oracle RAC.

The agent supports EMC MirrorView configured in the synchronous or asynchronous modes.

In asynchronous mode, you can replicate either individual LUNs or replicate consistency groups. MirrorView can also replicate LUNs or metaLUNs. In synchronous mode, you can replicate individual LUNs but cannot replicate consistency groups.

See the following Technical Support TechNote for the latest updates or software issues for this agent:

http://seer.entsupport.symantec.com/docs/282004.htm

Supported software

For information on the software versions that the agent for EMC MirrorView supports, see the Symantec Operations Readiness Tools (SORT) site: https://sort.veritas.com/agents.

Supported hardware

In environments using Storage Foundation for Oracle RAC, the arrays must support SCSI-3 persistent reservations.

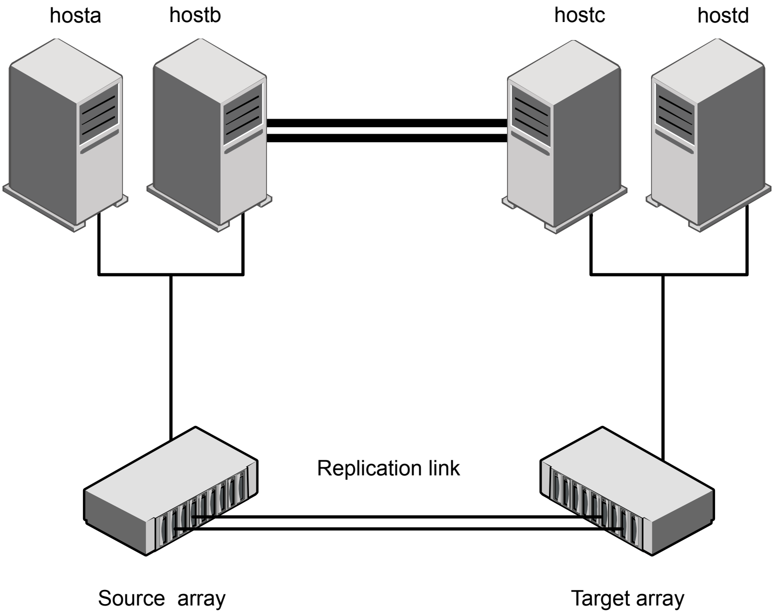
The Cluster Server agent for EMC Mirrorview supports Navisphere Secure CLI, naviseccli. The agent no longer supports the Java based CLI, navcli.jar.

To determine the supported versions of NaviCLI and FLARE code that are on CLARiiON arrays, consult the EMC hardware compatibility list.

Typical EMC MirrorView setup in a VCS cluster

Figure 1-1 displays a typical cluster setup in a MirrorView environment.

Figure 1-1 Typical clustering setup for the agent



Clustering in a MirrorView environment typically consists of the following hardware infrastructure:

- The source array (array1) has one or more hosts. The hosts have a direct connection to a CLARiiON array. The array contains the mirror that is the primary image and the direct connection uses either SCSI or Fibre Channel.
- The target array (array2) consists of one or more hosts. These hosts have a direct connection to another CLARiiON array. The array contains the mirror that is the secondary image and the connection uses either SCSI or Fibre Channel.
  The secondary image LUNs pairs with the mirrored LUNs in the source array. The target hosts and the array must be at a significant distance from the source side to survive a source-side disaster.
- Network heartbeating between the two data centers to determine their health could be LLT or TCP/IP.
  See “About cluster heartbeats” on page 24.
- In a replicated data cluster environment, all hosts are part of the same cluster. You must connect them with the dual and dedicated networks that support LLT.
  In a global cluster environment, you must attach all hosts in a cluster to the same CLARiiON array.

- In parallel applications like Storage Foundation for Oracle RAC, all hosts that are attached to the same array must be part of the same GAB membership. Storage Foundation for Oracle RAC is supported with Mirrorview only in a global cluster environment and not in a replicated data cluster environment.

EMC MirrorView agent functions

The Cluster Server agent for EMC MirrorView monitors and manages the state of replicated CLARiiON LUNs attached to VCS nodes. Agent functions bring resources online, take them offline, and perform different monitoring actions.

The agent performs the following functions:

Table 1-1 Agent functions

Function	Description
online	Creates a lock file on the local host. This lock indicates that the resource is online and makes the mirrors available for the application to use. The agent performs specific actions depending on the state of the mirrors. See “About the MirrorView agent’s online function” on page 13.
offline	Removes the lock file on the local host.
monitor	Verifies that the lock file exists. If the lock file exists, the monitor entry point reports the status of the resource as online. If the lock file does not exist, the monitor entry point reports the status of the resource as offline.
open	Prevents potential concurrency violation if the service group fails over to another node. **Note:** The agent does not remove the lock file if the agent was started after the `hastop -force` command.
clean	Removes the lock file.
info	The info function gives the information about the mirrors (in case of synchronous mode of replication). It also gives information about the mirrors/groups in case of asynchronous mode of replication. It uses the `-sync listsyncprogress` and `-async -list` or `-async listgroups` commands to get this information.
resync	Performs a resynchronization action.

Table 1-1 Agent functions *(continued)*

Function	Description
PreSwitch	Ensures that the remote site cluster can come online during a planned failover within a GCO configuration. The VCS engine on the remote cluster invokes the PreSwitch action on all the resources of the remote site during a planned failover using the `hagrp -switch` command. For this, the PreSwitch attribute must be set to 1. The option `-nopre` indicates that the VCS engine must switch the servicegroup regardless of the value of the PreSwitch service group attribute. If running the PreSwitch action fails, the failover should not occur. This minimizes the application downtime and data loss.
addArrayUser	This action entry point creates the user security file for the `naviseccli` command.

About the MirrorView agent’s online function

The agent's online operation performs specific actions depending on the state of the mirrors.

Synchronous state

If the state of all local mirrors is MIRRORED, the agent creates a lock file on the local host. This lock indicates that the resource is online and makes the mirrors available for the application to use.

If one or more mirrors are not in the MIRRORED state, the agent runs the `NaviCLI` command. With this command, the agent brings them into the MIRRORED state, which enables the application to use them.

- For secondary images in the synchronized state, the agent runs the `mirror -sync -promoteimage` command to promote the remote mirror. This command also converts the current primary to secondary.
- For secondary images in the CONSISTENT state, the agent waits to check if the image has transitioned to the SYNCHRONIZED state.
- If the images have transitioned to the SYNCHRONIZED state, the agent then runs the `mirror -sync -promoteimage` command to promote the remote mirror. This command also converts the current primary to secondary.
- If the image has not transitioned to the SYNCHRONIZED state, the agent checks if the remote array is accessible. If the remote array is accessible, then this condition indicates link failure—the image would be in a fractured condition.

In case of fracture:

- If the SplitTakeover attribute is set to 1, the agent forcibly promotes the secondary image.
- If the SplitTakeover attribute is set to 0, the agent does not try to promote the secondary image forcibly, and becomes the administrator's decision.

Asynchronous state

You can configure the online function for either consistency groups or mirrors in asynchronous mode.

Consistency groups

If the state of the group is SYNCHRONIZED, the agent creates a lock file on the local host to indicate that the resource is online. This lock makes the LUNs available for the application to use.

If one or more mirrors are not in the MIRRORED state, the agent checks to see if the remote array is accessible.

- If the remote array is not accessible, then the agent checks the value of the SplitTakeover attribute before proceeding with any further actions.
- If the SplitTakeover attribute is set to 1, the agent forcibly promotes the secondary image.
- If the SplitTakeover attribute is set to 0, the agent does not try to promote the secondary image forcibly, and becomes the administrator's decision.
- If the remote array is accessible, then the agent runs the `mirror -async -promotegroup` command to promote the remote group.
- In case of a successful promotegroup operation, the operation also converts the current primary to secondary.
- If the promotegroup operation is not successful, then the agent initiates a synchronization.
  The agent periodically checks if the group is SYNCHRONIZED. After a successful synchronization, the agent promotes the group using the `mirror -async -promotegroup` command. If the synchronization is not successful, the agent times out.

Mirrors

If the state of all local mirrors is MIRRORED, the agent creates a lock file on the local host. The lock indicates that the resource is online and makes the mirrors available for the application to use.

If one or more mirrors are not in the MIRRORED state, the agent checks to see if the remote array is accessible.

- If the remote array is not accessible, then the agent checks the value of the SplitTakeover attribute before proceeding with any further actions.
- If the SplitTakeover attribute is set to 1, the agent forcibly promotes the secondary image.
- If the SplitTakeover attribute is set to 0, the agent does not try to promote the secondary image forcibly, and becomes the administrator’s decision.
- If the remote array is accessible, then the agent runs the `mirror -async -promoteimage` command to promote the remote mirrors.
- A successful promoteimage operation converts the current primary to secondary.

  If the promoteimage operation is not successful, then the agent initiates a synchronization.

  The agent periodically checks if the group is SYNCHRONIZED. After a successful synchronization, the agent promotes the secondary mirror using the `mirror -async -promoteimage` command. If the synchronization is not successful, the agent times out.

Chapter 2

Installing and removing the agent for EMC MirrorView

This chapter includes the following topics:

- Before you install the agent for EMC MirrorView
- Installing the agent for EMC MirrorView
- Upgrading the agent for EMC MirrorView
- Removing the agent for EMC MirrorView

Before you install the agent for EMC MirrorView

Before you install the Cluster Server agent for EMC MirrorView, ensure that you install and configure the VCS on all nodes in the cluster.

Set up replication and the required hardware infrastructure. For information about setting up Oracle RAC environment, refer to the *Storage Foundation for Oracle RAC Configuration and Upgrade Guide*.

See “Typical EMC MirrorView setup in a VCS cluster” on page 10.

Installing the agent for EMC MirrorView

You must install the EMC MirrorView agent on each node in the cluster. In global cluster environments, install the agent on each node in each cluster.

These instructions assume that you have already installed VCS or SF for Oracle RAC.

To install the agent in a VCS environment

1. Download the Agent Pack from the Symantec Operations Readiness Tools (SORT) site: https://sort.veritas.com/agents.

   You can download the complete Agent Pack tar file or the individual agent tar file.

2. Uncompress the file to a temporary location, say /tmp.

3. If you downloaded the complete Agent Pack tar file, navigate to the directory containing the package for the platform running in your environment.

AIX	`cd1/aix/vcs/replication/mirrorview_agent/agent_version/pkgs/`
Linux	`cd1/linux/generic/vcs/replication/mirrorview_agent/agent_version/rpms/`
Solaris	`cd1/solaris/dist_arch/vcs/replication/mirrorview_agent/agent_version/pkgs/`

   If you downloaded the individual agent tar file, navigate to the pkgs directory (for AIX, and Solaris), or the rpms directory (for Linux).

4. Log in as a superuser.

5. Install the package.

   AIX

   ```
   # installp -ac -d VRTSvcsm.rte.bff VRTSvcsm.rte
   ```

   Linux

   ```
   # rpm -ihv \
   VRTSvcsm-AgentVersion-Linux_GENERIC.noarch.rpm
   ```

   Solaris

   ```
   # pkgadd -d . VRTSvcsm
   ```

Upgrading the agent for EMC MirrorView

You must upgrade the agent on each node in the cluster.

To upgrade the agent software

1. Save the VCS configuration and stop the VCS engine.

   ```
   # haconf -dump -makero
   # hastop -all -force
   ```

2. Remove the agent from the node.

   See "Removing the agent for EMC MirrorView" on page 18.

3. Delete the file /etc/VRTSvcs/conf/config/MirrorviewTypes.cf.
4. Install the current version of the agent.

   See "Installing the agent for EMC MirrorView" on page 16.

5. Copy the file MirrorviewTypes.cf from the directory /etc/VRTSvcs/conf/ to the /etc/VRTSvcs/conf/config directory.
6. Repeat step 2 through step 5 on each node.
7. From a node in the cluster, edit your configuration file /etc/VRTSvcs/conf/config/main.cf.

   Configure the new attributes, if applicable.

8. Verify the configuration by running the following command:

   ```
   # hacf -verify config
   ```

9. Start VCS on local node first.
10. Start VCS on other nodes.

Removing the agent for EMC MirrorView

Before you attempt to remove the agent, make sure the application service group is not online.

You must remove the HTC agent from each node in the cluster.

To remove the agent, type the following command on each node. Answer prompts accordingly:

AIX	`# installp -u VRTSvcsm.rte`
Linux	`# rpm -e VRTSvcsm`
Solaris	`# pkgrm VRTSvcsm`

Configuring the agent for EMC MirrorView

This chapter includes the following topics:

- Configuration concepts for the EMC MirrorView agent
- Before you configure the agent for EMC MirrorView
- Configuring the agent for EMC MirrorView

Configuration concepts for the EMC MirrorView agent

Review the resource type definition and the attribute definitions for the agent.

Resource type definition for the MirrorView agent

The resource type definition defines the agent in VCS.

```
type MirrorView (
    static keylist SupportedActions = { resync, PreSwitch, addArrayUser }
    static int MonitorInterval = 300
    static int NumThreads = 1
    static int OfflineMonitorInterval = 0
    static int RestartLimit = 1
    static str ArgList[] = { NaviCliHome, LocalArraySPNames,
    RemoteArraySPNames, Mode, GrpName, MirNames, SplitTakeover }
    str NaviCliHome = "/opt/Navisphere/bin"
    str LocalArraySPNames[]
    str RemoteArraySPNames[]
    str Mode
    str GrpName
    str MirNames[]
    int SplitTakeover
    temp str VCSResLock
    )
```

Attribute definitions for the MirrorView agent

Review the description of the agent attributes.

Required attributes

You must assign values to the following required attributes:

Table 3-1 Required attributes

Attribute	Description
NaviCliHome	NaviCLI installation directory `"/opt/Navisphere/bin"` Type-Dimension: string-scalar
LocalArraySPNames	The list of storage processors within the array to which the local hosts are connected. Can be names or IP addresses. Type-Dimension: string-vector
RemoteArraySPNames	The list of storage processors within the array to which the remote hosts are connected. Can be names or IP addresses. Type-Dimension: string-vector
RemoteArrayName	The name of the remote CLARiiON array that connect to the remote hosts. Type-Dimension: string-vector
Mode	The replication mode, which is either: `sync` or `async`. Type-Dimension: string-scalar
GrpName	The name of the consistency group to which the mirrors belong. This function applies with only one mode: `async`or `sync`. Type-Dimension: string-scalar

Table 3-1 Required attributes *(continued)*

Attribute	Description
MirNames	This function specifies the mirrors with only one replication mode: `sync` or `async`. Type-Dimension: string-vector
SplitTakeover	This integer indicates whether VCS should forcefully promote a secondary to a primary. In case of a link-failure between the two arrays, the state of the mirror remains consistent or out of sync. Under such circumstances, if the application has to failover—due to disaster or user-driven action—mirrors are not in a SYNCHRONIZED state. If the value of the SplitTakeOver attribute is 1: ■ The agent fails over when it discovers link failures ■ The agent determines that mirrors are out of sync If the value of the attribute is 0, agent does not fail over and the administrator must determine what to do. Type-Dimension: boolean-scalar

Optional attributes

Assign values to the following optional attributes.

Table 3-2 Optional attributes

Attribute	Description
LocalArraySecurityFilePath	The absolute path to the NaviSecCli security file, which contains credentials for the local array. If you do not assign a value to this attribute, the agent assumes that the security file is placed in the $HOME of the root user. Type-Dimension: string-scalar
RemoteArraySecurityFilePath	The absolute path to the NaviSecCli security file containing credentials for the remote array. If you do not assign a value to this attribute, the agent assumes that the security file is placed in the $HOME of the root user. Type-Dimension: string-scalar

Internal attribute

Do not modify internal attributes. The MirrorView agent currently supports the following internal attribute:

Table 3-3 Internal attribute

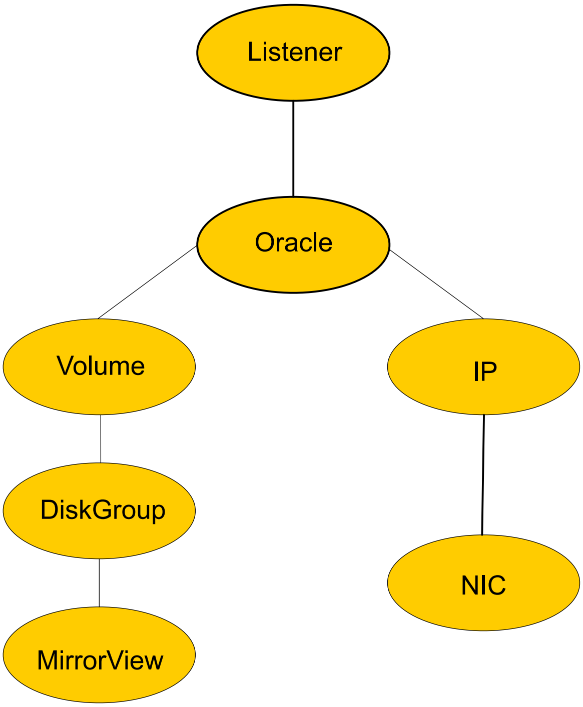
Attribute	Description
VCSResLock	This agent uses this attribute to guarantee serialized management in case of a parallel application. Do not modify this value. Type-Dimension: temporary-string

Sample configuration for the MirrorView agent

Figure 3-1 shows a VCS service group that has a resource of type MirrorView.

The DiskGroup resource depends on the MirrorView resource.

Figure 3-1 Dependency tree



You can configure a resource of type MirrorView in the `main.cf` file.

In this example, the resource is configured for asynchronous mode and consistency groups.

```
MirrorView mir (
    NaviCliHome = "/opt/Navisphere/bin"
    LocalArraySPNames = { "Local_SP1_IP", "Local_SP2_IP" }
    RemoteArraySPNames = { "Remote_SP1_IP", "Remote_SP2_IP" }

    Mode = async
    GrpName = async_consistency_grp1

    SplitTakeover = 0
    )
```

In this example, the resource is configured for synchronous mode and consistency groups.

```
MirrorView mir (
    NaviCliHome = "/opt/Navisphere/bin"
    LocalArraySPNames = { "Local_SP1_IP", "Local_SP2_IP" }
    RemoteArraySPNames = { "Remote_SP1_IP", "Remote_SP2_IP" }

    Mode = sync
    GrpName = sync_consistency_grp1

    SplitTakeover = 0
    )
```

If you want to configure the resource in synchronous mode and specify the individual mirror names, configure the MirNames attribute, instead of the GrpNames attribute, as follows:

```
    Mode = sync
    MirNames = { "sync_mir1", "sync_mir2" }
    GrpName = ""
```

If you want to configure the resource in asynchronous mode and specify the individual mirror names, configure the MirNames attribute, instead of the GrpNames attribute, as follows:

```
    Mode = async
    MirNames = { "async_mir1", "async_mir2" }
    GrpName = ""
```

Before you configure the agent for EMC MirrorView

Before you configure the agent, review the following information:

- Verify that you have installed the agent on all systems in the cluster.
- Verify the hardware setup for the agent.
  See "Typical EMC MirrorView setup in a VCS cluster" on page 10.
- Make sure that the cluster has an effective heartbeat mechanism in place.
  See "About cluster heartbeats" on page 24.
  See "About preventing split-brain" on page 25.
- Set up system zones in replicated data clusters.
  See "About configuring system zones in replicated data clusters" on page 24.

About cluster heartbeats

In a replicated data cluster, ensure robust heartbeating by using dual, dedicated networks over which the Low Latency Transport (LLT) runs. Additionally, you can configure a low-priority heartbeat across public networks.

In a global cluster, VCS sends ICMP pings over the public network between the two sites for network heartbeating. To minimize the risk of split-brain, VCS sends ICMP pings to highly available IP addresses. VCS global clusters also notify the administrators when the sites cannot communicate.

Heartbeat loss may occur due to the failure of all hosts in the primary cluster. In such a scenario, a failover may be required even if the array is alive. In any case, a host-only crash and a complete site failure must be distinguished. In a host-only crash, only the ICMP heartbeat signals a failure by an SNMP trap. No cluster failure notification occurs because a surviving heartbeat exists. This trap is the only notification to fail over an application.

About configuring system zones in replicated data clusters

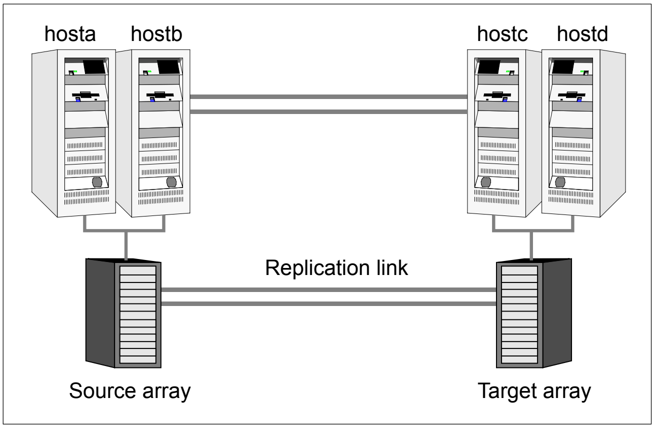
In a replicated data cluster, you can prevent unnecessary MirrorView failover or failback by creating system zones. VCS attempts to fail over applications within the same system zone before failing them over across system zones.

Configure the hosts that are attached to an array as part of the same system zone to avoid unnecessary failover.

Figure 3-2 depicts a sample configuration where hosta and hostb are in one system zone, and hostc and hostd are in another system zone.

Use the SystemZones attribute to create these zones.

Figure 3-2 Example system zone configuration



Modify the SystemZones attribute using the following command:

```
# hagrp -modify <group_name> SystemZones <system_name> <zone_number>
```

Global clusters do not require system zones because failover occurs on a remote cluster if all local targets have been exhausted.

As long as a secondary image is available, MirrorView sends the writes to the secondary image immediately in synchronous mode. It does so periodically in asynchronous mode.

If the period is too long, you can perform synchronization using the resync action. The supported resync action is defined in the MirrorView resource type.

About preventing split-brain

Split-brain occurs when all heartbeat links between the primary and secondary hosts are cut. In this situation, each side mistakenly assumes that the other side is down. You can minimize the effects of split-brain by ensuring that the cluster heartbeat links pass through a similar physical infrastructure as the replication links. When you ensure that both pass through the same infrastructure, if one breaks, so does the other.

Sometimes you cannot place the heartbeats alongside the replication links. In this situation, a possibility exists that the cluster heartbeats are disabled, but the replication link is not. A failover transitions the original source to target and target to source. In this case, the application faults because its underlying volumes become write-disabled, causing the service group to fault. VCS tries to fail it over to another host, causing the same consequence in the reverse direction. This phenomenon continues until the group comes online on the final node. You can avoid this situation by setting up your infrastructure such that loss of heartbeat links also mean the loss of replication links.

Configuring the agent for EMC MirrorView

You can configure clustered applications in a disaster recovery environment by:

- Converting their LUNs to CLARiiON LUNs
- Synchronizing the mirrors
- Adding the EMC MirrorView agent to the service group

After configuration, the application service group must follow the dependency diagram.

See “Sample configuration for the MirrorView agent” on page 22.

Note: You must not change the replication state of devices from primary to secondary and from secondary to primary, outside of a VCS setup. The agent for EMC MirrorView fails to detect a change in the replication state if the role reversal is done externally and RoleMonitor is disabled.

Configuring the agent manually in a global cluster

Configuring the agent manually in a global cluster involves the following tasks:

To configure the agent in a global cluster

1. Start Cluster Manager (Java Console) and log on to the cluster.
2. If the agent resource type (Mirrorview) is not added to your configuration, add it. From the Cluster Explorer **File** menu, choose **Import Types**, and select:

   /etc/VRTSvcs/conf/MirrorviewTypes.cf
3. Click **Import**.
4. Save the configuration.
5. Add a resource of type Mirrorview at the bottom of the service group.
6. Configure the attributes of the Mirrorview resource.
7. If the service group is not configured as a global service group, configure the service group using the Global Group Configuration Wizard.

   Refer to the *Cluster Server Administrator's Guide* for more information.

8. Change the ClusterFailOverPolicy attribute from the default, if necessary. Symantec recommends keeping the default, which is Manual, to minimize the chance of failing over on a split-brain.
9. Repeat step 5 through step 8 for each service group in each cluster that uses replicated data.

Configuring the agent manually in a replicated data cluster

Configuring the agent manually in a replicated data cluster involves the following tasks:

To configure the agent in a replicated data cluster

1. Start Cluster Manager and log on to the cluster.
2. If the agent resource type (Mirrorview) is not added to your configuration, add it. From the Cluster Explorer **File** menu, choose **Import Types** and select:

   /etc/VRTSvcs/conf/MirrorviewTypes.cf
3. Click **Import**.
4. Save the configuration.
5. In each service group that uses replicated data, add a resource of type Mirrorview at the bottom of the service group.
6. Configure the attributes of the Mirrorview resource.
7. Set the SystemZones attribute for the service group to reflect which hosts are attached to the same array.

Executing the addArrayUser action

Executing the addArrayUser action is a critical step in configuring the EMC MirrorView agent in both global clusters and replicated data clusters.

The MirrorView agent runs under the system user context. You must execute the addArrayUser action to create the required security files, thus allowing you as a system user to run commands on both local and remote arrays.

To execute the addArrayUser action

1. Right click the mirrorview resource and select **Actions**.
2. From the list of actions, select **addArrayUser**.
3. Select the system on which to execute the action.

4. From the list of action arguments, click the plus button to add a new argument.

   You can specify the credentials for the local and the remote CLARiiON arrays in the arguments as follows:

   ```
   localuser=LocalArrayUserName
   localpass=LocalArrayPassword
   remoteuser=RemoteArrayUserName
   remotepass=RemoteArrayPassword
   ```

   You can use this action to specify the credentials of the local array or remote array or both. You can also use the following two keys to specify the path to the security files for the arrays:

   ```
   localsecfilepath=LocalArraySecurityFilePath
   remotesecfilepath=RemoteArraySecurityFilePath
   ```

   If the security file path option is specified, this action also updates the appropriate resource attributes (LocalArraySecurityFilePath and/or RemoteArraySecurityFilePath) so that the subsequent naviseccli commands that the agent issues may succeed.

   Note: Use the security file path option only if you do not want the security files to be created in the default location.

5. Click **OK** to execute the action.
6. Review the results of the action and click **OK**.
7. Repeat these steps for all nodes that need to access the arrays.

Managing and testing clustering support for EMC MirrorView

This chapter includes the following topics:

- Typical test setup for the EMC MirrorView agent
- Testing service group migration
- Testing host failure
- Performing a disaster test
- Performing the failback test
- Failure scenarios for EMC MirrorView

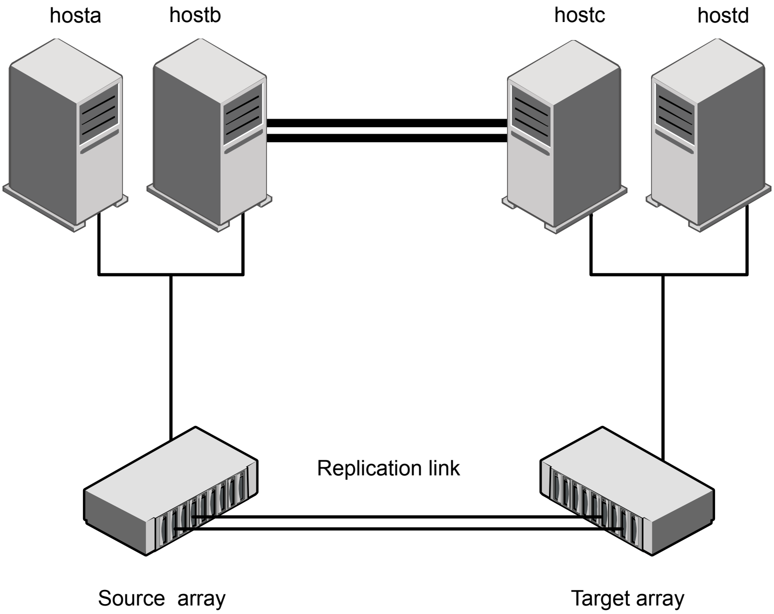
Typical test setup for the EMC MirrorView agent

A typical test environment includes the following characteristics:

- Two hosts (hosta and hostb) are attached to the source CLARiiON array.
- Two hosts (hostc and hostd) are attached to the target CLARiiON array.
- The application runs on hosta and devices in the local array are read-write enabled in the SYNCHRONIZED state.
- A replicated data cluster has two dedicated heartbeat links.
  A global cluster has one network heartbeat.

Figure 4-1 depicts a typical test environment.

Figure 4-1 Typical test setup



Testing service group migration

Verify that the service group can migrate to different hosts in the cluster and across clusters.

To perform the service group migration test

1 In the Cluster Explorer configuration tree, under the **Service Groups** tab, right-click the service group.

  Migrate the service group to a host that is attached to the same array.

2 Click **Switch To** and click the system that is attached to the same array (hostb) from the menu.

  The service group comes online on hostb and local image remains in the MIRRORED state.

3 In the **Service Groups** tab of the Cluster Explorer configuration tree, right-click the service group.

  Migrate the service group to a host that is attached to a different array.

4 Click **Switch To**, and click the system that is attached to another array (hostc) from the menu.

The service group comes online on hostc and the role of the images there transition to primary.

Accumulate dirty tracks on the new source-side and update them back on the target:

```
hares -action mirrorview_res_name resync -sys hostc
```

The variable *mirrorview_res_name* represents the name of the MirrorView resource.

5 In the **Service Groups** tab of the Cluster Explorer configuration tree, right-click the service group.

After the devices transition to a source SYNCHRONIZED state, migrate the service group back to its original host.

6 Click **Switch To** and click the system on which the group was initially online (hosta).

The group comes online on hosta. The devices return to the RW/SYNCINPROG state at the array that is attached to hosta and hostb, and then eventually transition to the SYNCHRONIZED state.

Testing host failure

In this scenario, the host where the application runs is lost. Eventually, all the hosts in the system zone or cluster are lost.

To perform the host failure test

1 Halt or shut down the host where the application runs (hosta).

The service group fails over to hostb and devices are in the SYNCHRONIZING state.

2 Halt or shut down hostb.

In a replicated data cluster, the group fails over to hostc or hostd depending on the FailOverPolicy attribute in the cluster.

In a global cluster, a cluster down alert appears and gives you the opportunity to fail over the service group manually.

In both environments, the role of the devices changes from secondary to primary and starts on the target host.

3 Power on the two hosts that were shut down.

4 Switch the service group to its original host when VCS starts.

   Do the following:

   - In the **Service Groups** tab of the Cluster Explorer configuration tree, right-click the service group.
   - Click **Switch To** and click the system on which the service group was initially online (hosta).
     The service group comes online on hosta and devices transition to the SYNCHRONIZING state and then to the SYNCHRONIZED state.

Performing a disaster test

Test how robust your cluster is in case of a disaster.

To perform a disaster test

1 Shut down all hosts on the source side and shut down the source array.

   If you cannot shut down the source array, change the value of the RemoteArraySPNames in the target side to non-existent names and IP addresses. This action mimics a disaster scenario from the target's point of view.

2 In a replicated data cluster, the service group fails over to hostc or hostd in the following conditions:

   - All devices were originally in the SYNCHRONIZED state.
   - No synchronization was in progress at the time of disaster.

3 In a global cluster, the administrator is notified of the failure. The administrator can then initiate the failover.

Performing the failback test

You can set up your cluster for a failback test.

The failback test verifies the application can fail back to its original host after a failover to a remote site.

To perform the failback test for asynchronous mode with Consistency groups

1 Remove all the mirrors form the consistency group on the old primary.

2 Destroy the consistency group on the old primary.

3 Forcefully destroy the remote mirrors on the old primary.

4. Remove the LUNs from the storage group on the old primary.
5. Remove the mirrors from the consistency group on the new primary.
6. Add secondary images to each of the remote mirrors on the new primary.
7. Add the mirrors into the consistency group on the new primary.

   Between step 5 and step 7, the LUNs become vulnerable to data corruption. For example, if one of the LUNs has sustained hardware damage and failed.

   During this window, the mirrors are not a part of the consistency group. The writes to other mirrors that were a part of the consistency group are not stopped. This situation could result in data corruption.
8. Add the LUNs, where the secondary image resides, into the appropriate storage group on the old primary.

To perform the failback test for synchronous and asynchronous mode with Individual mirrors

1. Forcefully destroy the remote mirrors on the old primary.
2. Remove the LUNs from the storage group on the old primary.
3. Add secondary images to each of the remote mirrors on the new primary.
4. Add the LUNs, where the secondary image resides, into the appropriate storage group on the old primary.

   In either of the modes, the original contents of the old primary are lost.

Failure scenarios for EMC MirrorView

Review the failure scenarios and agent behavior in response to failure.

Site disaster

In a total site failure, all hosts and the array are completely disabled, either temporarily or permanently.

In a replicated data cluster, site failure is detected the same way as a total host failure, that is, the loss of all LLT heartbeats.

In a global cluster, VCS detects site failure by the loss of all configured heartbeats.

A total disaster renders the devices on the surviving array in the FRACTURED state. If the SplitTakeover attribute is set to its default value of 1, the online entry point runs the ‘promote’ operation. If the attribute is set to 0, no takeover occurs and the online entry point times out and faults.

The online entry point detects whether any synchronization was in progress when the source array was lost. Since the target devices are inconsistent until the synchronization completes, the agent does not write-enable the devices, but it times out and faults. You must restore consistent data from a snapshot or tape backup.

All host or all application failure

Even if both arrays are operational, the service group fails over in the following conditions:

- All hosts on the source CLARiiON side are disabled.
- The application cannot start successfully on any source host.

In replicated data cluster environments, the failover can be automatic, whereas in global cluster environments failover requires user confirmation by default.

In replicated data cluster environments, site failure is detected the same way as a total host failure, that is, the loss of all LLT heartbeats. This type of failure is communicated by the VCS engine to the other site.

In both replicated data cluster and global cluster environments, multiple service groups can fail over in parallel.

Replication link failure

Before the MirrorView takes any action, it waits for the synchronization to complete in the following situations:

- The two arrays are healthy and the link that failed is restored.
- A failover is initiated while synchronization is in progress.

After the synchronization completes, the MirrorView runs the promote operation.

If the agent times out before the synchronization completes, the resource faults.

If the SplitTakeover attribute is set to 0, the agent does not attempt a promote operation, but it times out and faults. If you write-enable the devices manually, the agent can come online after it is cleared.

Split-brain in a MirrorView environment

You must resynchronize the volumes manually by using any of the following commands:

```
naviseccli -h array_ip mirror -async/-sync -syncimage
naviseccli -h array_ip mirror -async/-sync -syncgroup
```

In a global cluster, you can confirm the failure before failing over the service groups. You can check with the site administrator to identify the cause of the failure. If a fail over mistakenly occurs, the situation is similar to the replicated data cluster case. However, when the heartbeat is restored, VCS does not stop HAD at either site. VCS forces you to choose which group to take offline. You must resynchronize the data manually.

Chapter 5

Setting up a fire drill

This chapter includes the following topics:

- About fire drills
- Considerations for using MirrorView and SnapView together
- About the MirrorViewSnap agent
- Before you configure the fire drill service group
- Configuring the fire drill service group
- Verifying a successful fire drill
- Sample configuration for a fire drill service group

About fire drills

A fire drill procedure verifies the fault-readiness of a disaster recovery configuration. This procedure is done without stopping the application at the primary site and disrupting user access.

A fire drill is performed at the secondary site using a special service group for fire drills. The fire drill service group is identical to the application service group, but uses a fire drill resource in place of the replication agent resource. The fire drill service group uses a copy of the data that is used by the application service group.

In clusters employing EMC MirrorView, the MirrorViewSnap resource manages the replication relationship during a fire drill.

Bringing the fire drill service group online demonstrates the ability of the application service group to come online at the remote site when a failover occurs.

The MirrorViewSnap agent supports fire drills for storage devices that are managed using Veritas Volume Manager, which is a component of Storage Foundation.

Considerations for using MirrorView and SnapView together

MirrorView is an EMC software application that maintains a copy or image of a logical unit (LUN) at a separate location. This copy or image is a provision for disaster recovery. You can use this image in the event that a disaster disables the production image. The production image is called the primary image; the copy image is called the secondary image.

SnapView is a storage-system-based software application that enables you to create a copy of a LUN by using either clones or snapshots. A clone is an actual copy of a LUN and takes time to create, depending on the size of the source LUN. A clone is a full copy. A snapshot is a virtual point-in-time copy of a LUN and takes only seconds to create. A snapshot uses a copy-on-write principle.

Fire drills use SnapView with MirrorView. The VCS MirrorView fire drill agent, MirrorViewSnap, does not support clones because of the following considerations:

- If a LUN is a MirrorView primary or secondary image, you cannot create a clone group for that image. If a LUN is a member of a clone group as the source or clone, it cannot serve as a MirrorView primary or secondary image.
- If the MirrorView Synchronous option is installed, you can create a snapshot of the primary or secondary image. However, Symantec recommends that you take a snapshot of a secondary image only if the state of the image is either SYNCHRONIZED or CONSISTENT. If the image is in the SYNCHRONIZING state or in the OUT-OF-SYNC state, the snapshot data is not useful.
- If the MirrorView Asynchronous option is installed, you can create a snapshot of the primary or secondary image. However, Symantec recommends that you take a snapshot of a secondary image only if the last update started has completed successfully. If the update did not complete successfully because the image is fractured or the update is still in progress, the snapshot data is not useful.

The VCS MirrorView fire drill agent, MirrorViewSnap, does not support clones because of these considerations.

About the MirrorViewSnap agent

The MirrorViewSnap agent is the fire drill agent for EMC MirrorView.

The agent manages the replication relationship between the source and target arrays when running a fire drill. Configure the MirrorViewSnap resource in the fire drill service group, in place of the Mirrorview resource.

MirrorViewSnap agent functions

The MirrorViewSnap agent performs the following functions:

Table 5-1 Agent functions

Function	Description
online	Takes a snapshot of the LUNs in the mirror/consistency group and make it available for use. It performs the following steps: ■ Retrieves the state of the mirror/consistency group. If the mirror/consistency group is not in either SYNCHRONIZED/CONSISTENT state, the fire drill cannot continue. The agent logs an error and exits. ■ Creates the MirrorViewSnap object. ■ Takes the snapshot. In case of any error, the agent rolls back the changes for all the other mirrors. After the snapshot is successfully taken, the agent prepares the disk group for importation. ■ Creates a lock file and exits.
offline	Deletes the snapshot. If successful, the Offline entry point removes the lock file.
monitor	Verifies the existence of the lock file to make sure the resource is online.
clean	Deletes the snapshot. If successful, the Offline entry point removes the lock file.
open	If the lock file does not exist, the agent takes no action. If the lock file exists: ■ If any resources that depend on the MirrorViewSnap are online, the agent does not take any action. ■ If any resources that depend on the MirrorViewSnap are not online, the agent removes the lock file.

Resource type definition for the MirrorViewSnap agent

Following is the resource type definition for the MirrorViewSnap agent:

```
type MirrorViewSnap (
    static int MonitorInterval = 300
    static int NumThreads = 1
    static int OfflineMonitorInterval = 0
    static int OnlineTimeout = 600
    static int RestartLimit = 1
    stat    ic str ArgList[] = { StorageGrpName, TargetResName }
    str TargetResName
    str StorageGrpName
    temp str Responsibility
)
```

Attribute definitions for the MirrorViewSnap agent

To customize the behavior of the MirrorViewSnap agent, configure the following attributes:

Table 5-2 Agent attributes

Attribute	Description
TargetResName	Name of the resource managing the LUNs that you want to take snapshot of. Set this attribute to the name of the Mirrorview resource if you want to take a snapshot of replicated data. If the data is not replicated and if the NaviCLI version is 6.28 or earlier, set this attribute to the name of the DiskGroup resource. For example, in a typical Oracle setup, you might replicate data files and redo logs, but you may choose to avoid replicating temporary tablespaces. The temporary tablespace must still exist at the DR site and may be part of its own disk group. Type-Dimension: string-scalar
StorageGrpName	Name of the storage group that contains the snapshot. The host that runs the fire drill must be a part of this storage group. Otherwise, the fire drill fails. Type-Dimension: string-scalar
Responsibility	Do not modify. For internal use only. Used by the agent to keep track of resynchonizing snapshots. Type-Dimension: temporary string

Before you configure the fire drill service group

Before you configure the fire drill service group, ensure that the following pre-requisites are met:

- Make sure the application service group is configured with a Mirrorview resource.
- Make sure the infrastructure to take snapshots is properly configured between the source and target arrays.
- Make sure the MirrorView relationship is established.
- Reserve sufficient unallocated LUNs in the reserved LUN pool.
- Install and enable the SnapView license.
- Install the Navisphere CLI

Configuring the fire drill service group

On the secondary site, the initial steps create a fire drill service group that closely follows the configuration of the original application service group. The fire drill service group uses a point-in-time copy of the production data. Bringing the fire drill service group online on the secondary site demonstrates the ability of the application service group to fail over and come online at the secondary site, should the need arise.

See “Sample configuration for a fire drill service group” on page 42.

Creating the fire drill service group using Cluster Manager (Java Console)

This section describes how to use Cluster Manager (Java Console) to create the fire drill service group. After creating the fire drill service group, you must set the failover attribute to false so that the fire drill service group does not fail over to another node during a test.

To create the fire drill service group

1. Open the Cluster Manager (Java Console).
2. Log on to the cluster and click **OK**.
3. Click the **Service Group** tab in the left pane and click the **Resources** tab in the right pane.
4. Right-click the cluster in the left pane and click **Add Service Group**.
5. In the **Add Service Group** dialog box, provide information about the new service group.

- In Service Group name, enter a name for the fire drill service group.
- Select systems from the Available Systems box and click the arrows to add them to the Systems for Service Group box.
- Click **OK.**

To disable the AutoFailOver attribute

1. Click the **Service Group** tab in the left pane and select the fire drill service group.
2. Click the **Properties** tab in the right pane.
3. Click the **Show all attributes** button.
4. Double-click the **AutoFailOver** attribute.
5. In the **Edit Attribute** dialog box, clear the **AutoFailOver** check box.
6. Click **OK** to close the **Edit Attribute** dialog box.
7. Click the **Save and Close Configuration** icon in the toolbar.

Adding resources to the fire drill service group

Add resources to the new fire drill service group to recreate key aspects of the application service group.

To add resources to the service group

1. In Cluster Explorer, click the **Service Group** tab in the left pane, click the application service group and click the **Resources** tab in the right pane.
2. Right-click the resource at the top of the tree, select **Copy > Self and Child Nodes**.
3. In the left pane, click the fire drill service group.
4. Right-click the right pane, and click **Paste**.
5. In the **Name Clashes** dialog box, specify a way for the resource names to be modified, for example, insert an '_fd' suffix. Click **Apply**.
6. Click **OK**.

Configuring resources for fire drill service group

Edit the resources in the fire drill service group so they work properly with the duplicated data. The attributes must be modified to reflect the configuration at the remote site. Bringing the service group online without modifying resource attributes is likely to result in a cluster fault and interruption in service.

To configure the fire drill service group

1. In Cluster Explorer, click the **Service Group** tab in the left pane.
2. Click the fire drill service group in the left pane and click the **Resources** tab in the right pane.
3. Right-click the Mirrorview resource and click **Delete**.
4. Add a resource of type MirrorViewSnap and configure its attributes.
5. Right-click the resource to be edited and click **View > Properties View**. If a resource to be edited does not appear in the pane, click **Show All Attributes**.
6. Edit attributes to reflect the configuration at the remote site. For example, change the Mount resources so that they point to the volumes that are used in the fire drill service group.

Verifying a successful fire drill

Run the fire drill routine periodically to verify the application service group can fail over to the remote node.

To verify a successful fire drill

1. Bring the fire drill service group online on a node at the secondary site that does not have the application running.

   If the fire drill service group comes online, it action validates your disaster recovery configuration. The production service group can fail over to the secondary site in the event of an actual failure (disaster) at the primary site.
2. If the fire drill service group does not come online, review the VCS engine log for more information.
3. Take the fire drill offline after its functioning has been validated.

   Failing to take the fire drill offline could cause failures in your environment. For example, if the application service group fails over to the node hosting the fire drill service group, there would be resource conflicts, resulting in both service groups faulting.

Sample configuration for a fire drill service group

The sample configuration of a fire drill service group is identical to an application service group with a hardware replication resource. However, in a fire drill service group, the MirrorViewSnap resource replaces the Mirrorview resource.

You can configure a resource of type MirrorViewSnap in the main.cf file as follows:

```
group myapp (
SystemList = { thoribm102 = 0 }
ClusterList = { vcspri = 0, vcsdr = 1 }
Authority = 1
)
Application testApp (
StartProgram = "/testApp/start.sh"
StopProgram = "/testApp/stop.sh"
MonitorProgram = "/testApp/monitor.sh"
)
DiskGroup VM_mvsnapdg (
DiskGroup = mvsnapFD-Groupdg
)DiskGroup VM_nonrep_dg (
DiskGroup = nonrep_dg
)
MirrorView MV-mvtestApp (
LocalArraySPNames @thoribm102 = { cx700spj1, cx700spj2 }
RemoteArraySPNames @thoribm102 = { cx600c2, "10.180.66.138" }
Mode = async
MirNames = { AsyncMir1Aix, AsyncMir2Aix }
)
Mount Mount-testvol1 (
MountPoint = "/testdir/testvol1"
BlockDevice = "/dev/vx/dsk/mvsnapFD-Groupdg/testvol1"
FSType = vxfs
FsckOpt = "-y"
)
Mount Mount-testvol2 (
MountPoint = "/testdir/testvol2"
BlockDevice = "/dev/vx/dsk/mvsnapFD-Groupdg/testvol2"
FSType = vxfs
FsckOpt = "-y"
)
Mount Mount-testvol3 (
MountPoint = "/testdir/testvol3"
BlockDevice = "/dev/vx/dsk/mvsnapFD-Groupdg/testvol3"
FSType = vxfs
FsckOpt = "-y"
)
Mount Mount-testvol4 (
MountPoint = "/testdir/testvol4"
BlockDevice = "/dev/vx/dsk/mvsnapFD-Groupdg/testvol4"
FSType = vxfs
```

```
FsckOpt = "-y"
)
Mount Mount-vol01 (
MountPoint = "/nonreptestdir/vol01"
BlockDevice = "/dev/vx/dsk/nonrep_dg/vol01"
FSType = vxfs
FsckOpt = "-y"
)
Mount Mount-vol02 (
MountPoint = "/nonreptestdir/vol02"
BlockDevice = "/dev/vx/dsk/nonrep_dg/vol02"
FSType = vxfs
FsckOpt = "-y"
)
Mount-testvol1 requires VM_mvsnapdg
Mount-testvol2 requires VM_mvsnapdg
Mount-testvol3 requires VM_mvsnapdg
Mount-testvol4 requires VM_mvsnapdg
Mount-vol01 requires VM_nonrep_dg
Mount-vol02 requires VM_nonrep_dg
VM_mvsnapdg requires MV-mvtestApptestApp requires Mount-testvol1
testApp requires Mount-testvol2
testApp requires Mount-testvol3
testApp requires Mount-testvol4
testApp requires Mount-vol01
testApp requires Mount-vol02
// Fire drill group
group myapp_fd (
SystemList = { thoribm102 = 0 }
)
Application testApp_fd (
StartProgram = "/testApp/start.sh"
StopProgram = "/testApp/stop.sh"
MonitorProgram = "/testApp/monitor.sh"
)
DiskGroup VM_mvsnapdg_fd (
DiskGroup = mvsnapFD-Groupdg_fd
)
DiskGroup VM_nonrep_dg_fd (
DiskGroup = nonrep_dg_fd
)
MirrorViewSnap MV-dgtestApp_fd (
TargetResName = VM_nonrep_dg
```

```
StorageGrpName = SnapViewAix-SG
)
MirrorViewSnap MV-mvtestApp_fd (
TargetResName = MV-mvtestApp
StorageGrpName = SnapViewAix-SG
)
Mount Mount-testvol1_fd (
MountPoint = "/testdir/testvol1"
BlockDevice = "/dev/vx/dsk/mvsnapFD-Groupdg_fd/testvol1"
FSType = vxfs
FsckOpt = "-y"
)
Mount Mount-testvol2_fd (
MountPoint = "/testdir/testvol2"
BlockDevice = "/dev/vx/dsk/mvsnapFD-Groupdg_fd/testvol2"
FSType = vxfs FsckOpt = "-y"
)
Mount Mount-testvol3_fd (
MountPoint = "/testdir/testvol3"
BlockDevice = "/dev/vx/dsk/mvsnapFD-Groupdg_fd/testvol3"
FSType = vxfs
FsckOpt = "-y"
)Mount Mount-testvol4_fd (
MountPoint = "/testdir/testvol4"
BlockDevice = "/dev/vx/dsk/mvsnapFD-Groupdg_fd/testvol4"
FSType = vxfs
FsckOpt = "-y"
)
Mount Mount-vol01_fd (
MountPoint = "/nonreptestdir/vol01"
BlockDevice = "/dev/vx/dsk/nonrep_dg_fd/vol01"
FSType = vxfs
FsckOpt = "-y"
)
Mount Mount-vol02_fd (
MountPoint = "/nonreptestdir/vol02"
BlockDevice = "/dev/vx/dsk/nonrep_dg_fd/vol02"
FSType = vxfs
FsckOpt = "-y"
)
Mount-testvol1_fd requires VM_mvsnapdg_fd
Mount-testvol2_fd requires VM_mvsnapdg_fd
Mount-testvol3_fd requires VM_mvsnapdg_fd
```

```
Mount-testvol4_fd requires VM_mvsnapdg_fd
Mount-vol01_fd requires VM_nonrep_dg_fd
Mount-vol02_fd requires VM_nonrep_dg_fd
VM_mvsnapdg_fd requires MV-mvtestApp_fd
VM_nonrep_dg_fd requires MV-dgtestApp_fd
testApp_fd requires Mount-testvol1_fd
testApp_fd requires Mount-testvol2_fd
testApp_fd requires Mount-testvol3_fd
testApp_fd requires Mount-testvol4_fd
testApp_fd requires Mount-vol01_fd
testApp_fd requires Mount-vol02_fd
```

Index

A

- agent functions
  - addArrayUser 12
  - clean 12
  - info 12
  - monitor 12
  - offline 12
  - online 12
  - open 12
  - PreSwitch 12
  - resynch 12
- application failure 34
- attribute definitions 20

C

- cluster
  - heartbeats 24
- configuring
  - before 19
  - samples 22

D

- disaster test 32

F

- failback test 32
- failure scenarios
  - all application failure 34
  - all host failure 34
  - replication link failure 34
  - total site disaster 33
- fire drill
  - about 36
  - configuration wizard 40
  - MirrorViewSnap agent 37
  - running 42
  - service group for 40

H

- host failure 34

I

- installing the agent
  - AIX systems 16
  - Linux systems 16
  - Solaris systems 16
- internal attributes
  - VCSResLock 22

M

- migrating service group 30
- MirrorViewSnap agent
  - about 37
  - attribute definitions 39
  - operations 37
  - type definition 38
- MirrorViewSnap agent attributes
  - Responsibility 39

O

- Online function
  - consistency groups 13
  - mirrors 13
  - mirrors in asynchronous state 13
  - mirrors in synchronous state 13
- optional attributes
  - LocalArraySecurityFilePath 21
  - RemoteArraySecurityFilePath 21

R

- replication link failure 34
- required attributes
  - GrpName 20
  - LocalArraySPNames 20
  - MirNames 20
  - Mode 20
  - NaviCliHome 20
  - RemoteArrayName 20
  - RemoteArraySPNames 20
  - SplitTakeover 20

resource type definition
  MirrorViewSnap agent 38
Responsibility attribute 39

S

sample configuration\x0d 22
service group
  migrating 30
split-brain
  handling in cluster 25
  handling in clusters 34

T

testing
  disaster 32
  failback 32
testing failback
  asynchronous mode with consistency groups 32
  asynchronous with individual mirrors 33
  synchronous with individual mirrors 33
total site disaster 33
type definition
  MirrorViewSnap agent 38

U

uninstalling the agent
  AIX systems 18
  Linux systems 18
  Solaris systems 18